UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/785,143	02/16/2001	Steven Orodon Hobbs	2007.013000/TDM	4992

7590 02/05/2004

JONATHAN M. HARRIS
CONLEY, ROSE & TAYON
P.O. BOX 3267
HOUSTON, TX 77253-3267

EXAMINER	
FOWLKES, ANDRE R	
ART UNIT	**PAPER NUMBER**
2122	10

DATE MAILED: 02/05/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/785,143	HOBBS ET AL.	
	Examiner	Art Unit	
	Andre R. Fowlkes	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 30 January 2002.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-35 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-35 is/are rejected.

7)☒ Claim(s) 1 is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on 1/30/02 is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .	6) ☐ Other: .

DETAILED ACTION

1. Claims 1-35 are pending.

Drawings

2. Figures 1-3 should be designated by a legend such as --Prior Art-- because only that which is old is illustrated. See MPEP § 608.02(g). A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Specification

3. The disclosure is objected to because of the following informalities:

- "of a porton of the" should be --of a portion of the-- on p. 5 line 19;

- "may useful" should be --may be useful-- on p. 14 line 18;

Appropriate correction is required.

Claim Objections

4. Claim 1 is objected to because of the following informalities: "identifying each vector memory reference in the loop" should be -- identifying each vector memory reference in the loop; -- on line 3. Appropriate correction is required.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

Claims 1-4, 9-12, 17-19, 24-28 and 33-35 are rejected under 35 U.S.C. 102(b) as being anticipated by Carr et al. (Carr), "Compiler Optimizations for Improving Data Locality", 1994, ACM, p. 252-262.

As per claim 1, Carr discloses **identifying a loop in a program; identifying each vector memory reference in the loop; determining dependencies between vector memory references in the loop, including determining unidirectional and circular dependencies; and distributing the vector memory references into a plurality of detail loops, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop, and wherein the detail loops are ordered according to the unidirectional dependencies between the memory references** (p. 253 col. L lines 2-6, "applying compiler transformations based on data dependence (e.g., loop interchange, fusion, distribution, and tiling) to improve paging… In this paper, we … integrate optimizations for parallelism and memory", and p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) … (in a loop)", and p. 256

col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same (common detail) loop (and the detail loops must be ordered according to the unidirectional dependencies between the memory references)").

As per claim 2, the rejection of claim 1 is incorporated and further, Carr discloses **allocating a plurality of temporary storage areas within a cache and determining the size of each temporary storage area based on the size of the cache and the number of temporary storage areas** (p. 252 col. R lines 9-12, "loop ... distribution ... requires knowledge ... of the cache line size", and p. 252 col. R lines 14-15, "Knowledge of the cache size, associativity, and replacement policy is essential", and the optimization technique of loop distribution includes the allocation of a plurality of temporary storage areas within a cache and determining the size of each temporary storage area based on the size of the cache and the number of temporary storage areas).

As per claim 3, the rejection of claim 1 is incorporated and further, Carr discloses a **section loop including the plurality of detail loops** (p. 256 col. L line 48, "Loop distribution").

As per claim 4, the rejection of claim 1 is incorporated and further, Carr discloses **distributing the vector memory references into a plurality of detail loops further comprises distributing the vector memory references into a plurality of detail loops that each contain at least one vector memory reference that could benefit**

from cache management (p. 253 col. L lines 2-6, "applying compiler transformations based on data dependence (e.g., loop interchange, fusion, distribution, and tiling) to improve paging").

As per claim 9 Carr discloses **identifying a loop in a program; identifying each vector memory reference in the loop; determining dependencies between vector memory references in the loop; and distributing the vector memory references into a plurality of detail loops, wherein the vector memory references that have dependencies therebetween are included in a common detail loop** (p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) ... (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same loop").

As per claim 10, the rejection of claim 9 is incorporated and further, claim 10 is a method claim corresponding to method claim 2 and is rejected for the reasons set forth in the rejection of claim 2.

As per claim 11, the rejection of claim 9 is incorporated and further, claim 11 is a method claim corresponding to method claim 3 and is rejected for the reasons set forth in the rejection of claim 3.

As per claim 12, the rejection of claim 9 is incorporated and further, claim 12 is a method claim corresponding to method claim 4 and is rejected for the reasons set forth in the rejection of claim 4.

As per claim 17 Carr discloses **identifying a loop in a program; identifying each vector memory reference in the loop determining dependencies between vector memory references in the loop**(p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) ... (in a loop)"); **and distributing the vector memory references into a plurality of detail loops in response to cache behavior and the dependencies between the vector memory references in the loop** (p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers", and p. 252 col. R lines 14-15, "Knowledge of the cache size, associativity, and replacement policy (i.e. cache behavior) is essential").

As per claim 18, the rejection of claim 17 is incorporated and further, Carr discloses **determining dependencies between vector memory references in the loop, and wherein distributing the loop includes distributing the vector memory references into the plurality of detail loops, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop** (p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) ... (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same loop").

As per claim 19, the rejection of claim 17 is incorporated and further, Carr discloses **determining dependencies between vector memory references in the**

loop, and wherein distributing the loop includes distributing the vector memory references into the plurality of detail loops, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop (p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) … (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same loop").

As per claim 24 Carr discloses **a computer programmed to perform a method, comprising: identifying a loop in a program; identifying each vector memory reference in the loop determining dependencies between vector memory references in the loop; and distributing the vector memory references into a plurality of detail loops, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop** (p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) … (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same loop").

As per claim 25 Carr discloses a **program storage medium encoded with instructions that, when executed by a computer, perform a method, comprising: identifying a loop in a program; identifying each vector memory reference in the**

loop determining dependencies between vector memory references in the loop; and distributing the vector memory references into a plurality of detail loops, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop (p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) ... (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same loop").

Claim 26 is a compiler claim corresponding to method claim 1 and is rejected for the reasons set forth in the rejection of claim 1.

As per claim 27, the rejection of claim 26 is incorporated and further, claim 27 is a compiler claim corresponding to method claim 2 and is rejected for the reasons set forth in the rejection of claim 2.

As per claim 28, the rejection of claim 26 is incorporated and further, claim 28 is a compiler claim corresponding to method claim 3 and is rejected for the reasons set forth in the rejection of claim 3.

As per claim 33 Carr discloses a **method for reducing the likelihood of cache thrashing by software to be executed on a computer system having a cache, comprising: executing the software on the computer system; generating a profile indicating the manner in which the software uses the cache; identifying a portion of the software using the profile data that may experience cache thrashing; and**

modifying the identified portion of the softwar to reduce the likelihood of cache thrashing (p. 253 col. L lines 61-62, "data dependence (i.e. a portion of the software that may experience cache thrashing) (is identified) between two arrays ... (in a loop)", and p. 253 col. L lines 2-6, "compiler transformations (are applied) based on data dependence (e.g., loop interchange, fusion, distribution, and tiling) to improve paging (to reduce the likelihood of cache thrashing)... In this paper, we ... integrate optimizations for parallelism and memory").

As per claim 34, the rejection of claim 33 is incorporated and further, Carr discloses **modifying the identified portion of the software to reduce the likelihood of cache thrashing further comprises: identifying a loop in the identified portion of the software; identifying each vector memory reference in the identified loop; determining dependencies between the vector memory references in the identified loop of the software, including determining unidirectional and circular dependencies; and distributing the vector memory references into a plurality of detail loops, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop, and wherein the detail loops are ordered according to the unidirectional dependencies between the memory references** (p. 253 col. L lines 2-6, "applying compiler transformations based on data dependence (e.g., loop interchange, fusion, distribution, and tiling) to improve paging", and p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) ... (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same (detail) loop (and the detail loops must be ordered according to the unidirectional dependencies between the memory references)").

As per claim 35 Carr discloses a **method for reducing the likelihood of cache thrashing by software to be executed on a computer system having a cache, comprising: executing the software on the computer system; generating a profile indicating the manner in which the memory references of the software use the cache (** p. 253 col. L lines 2-6, "applying compiler transformations based on data dependence (profile) to improve paging**); identifying a first and second portion of the memory references based on the profile, wherein the first portion of the memory references may be experiencing cache thrashing; and distributing at least a portion of the first portion of the memory references into distinct loops, and placing at least the second portion of the memory references into the distinct loops** (p. 253 col. L lines 61-62, "data dependence (is determined) between two arrays (vector memory references) ... (in a loop)", and p. 256 col. L lines 48-51, "Loop distribution separates independent statements in a single loop into multiple loops with identical headers. To maintain the meaning of the original loop, statements in a recurrence (a cycle in the dependence graph) must be placed in the same loop").

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 5-8, 13-16, 20-23 and 29-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Carr et al. (Carr), "Compiler Optimizations for Improving Data Locality", 1994, ACM, p. 252-262 in view of Mahadevan et al. (Mahadevan) U.S. Patent No. 5,797,013.

As per claim 5, the rejection of claim 1 is incorporated and further, Carr doesn't explicitly disclose **inserting cache management instructions into at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory.**

However, Mahadevan, in an analogous environment, discloses **inserting cache management instructions into loops to control movement of data associated with the vector memory reference between a cache and main memory** (col. 6 lines 54-55, "(the compiler can) insert prefetches and effect other optimizations (cache management instructions) into the ... loop code").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Mahadevan into the system of Carr to have cache management instructions inserted into detail loops to control movement of data associated with the vector memory reference between a cache and main memory. The modification would have been obvious because one of ordinary skill in the art would want to compile the code using techniques that will maximize the efficiency of the compiled code's cache usage and therefore overall operation.

As per claim 6, the rejection of claim 1 is incorporated and further, Carr doesn't explicitly disclose **inserting prefetch instructions into at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory.**

However, Mahadevan, in an analogous environment, discloses **inserting prefetch instructions into loops to control movement of data associated with the vector memory reference between a cache and main memory** (col. 6 lines 54-55, "(the compiler can) insert prefetches and effect other optimizations into the ... loop code").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Mahadevan into the system of Carr to have prefetch instructions inserted into detail loops to control movement of data associated with the vector memory reference between a cache and main memory. The modification would have been obvious because one of ordinary skill in the art would want to compile the code using techniques that will maximize the efficiency of the compiled code's cache usage and therefore overall operation.

As per claim 7, the rejection of claim 1 is incorporated and further, Carr doesn't explicitly disclose **performing loop unrolling on at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory.**

However, Mahadevan, in an analogous environment, discloses **performing loop unrolling on loops to control movement of data associated with the memory reference between a cache and main memory** (col. 6 line 27, "the compiler unrolls loops").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Mahadevan into the system of Carr to have loop unrolling performed on at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory. The modification would have been obvious because one of ordinary skill in the art would want optimize the performance of the compiled code.

As per claim 8, the rejection of claim 1 is incorporated and further, Carr doesn't explicitly disclose **inserting at least one of a prefetch instruction and a cache management instruction into at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory, and performing loop unrolling on at least one of said detail**

loops to control movement of data associated with the vector memory reference between a cache and main memory.

However, Mahadevan, in an analogous environment, discloses **inserting a prefetch instruction and a cache management instruction into loops to control movement of data associated with the memory reference between a cache and main memory, and performing loop unrolling on loops to control movement of data associated with the memory reference between a cache and main memory** (col. 6 lines 54-55, "(the compiler can) insert prefetches and effect other optimizations (cache management instructions) into the ... loop code", and col. 6 line 27, "the compiler unrolls loops").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Mahadevan into the system of Carr to have the insertion of at least one of a prefetch instruction and a cache management instruction into at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory, and performing loop unrolling on at least one of said detail loops to control movement of data associated with the vector memory reference between a cache and main memory The modification would have been obvious because one of ordinary skill in the art would want to gain the performance advantages provided by using these optimization techniques in combination (Mahadevan, col. 6 line 22 – col. 7 line 29).

As per claim 13 , the rejection of claim 9 is incorporated and further claim 13 is a method claim corresponding to method claim 5 and is rejected for the reasons set forth in the rejection of claim 5.

As per claim 14 , the rejection of claim 9 is incorporated and further claim 14 is a method claim corresponding to method claim 6 and is rejected for the reasons set forth in the rejection of claim 6.

As per claim 15 , the rejection of claim 9 is incorporated and further claim 15 is a method claim corresponding to method claim 7 and is rejected for the reasons set forth in the rejection of claim 7.

As per claim 16 , the rejection of claim 9 is incorporated and further claim 16 is a method claim corresponding to method claim 8 and is rejected for the reasons set forth in the rejection of claim 8.

As per claim 20 , the rejection of claim 17 is incorporated and further claim 20 is a method claim corresponding to method claim 5 and is rejected for the reasons set forth in the rejection of claim 5.

As per claim 21 , the rejection of claim 17 is incorporated and further claim 21 is a method claim corresponding to method claim 6 and is rejected for the reasons set forth in the rejection of claim 6.

As per claim 22 , the rejection of claim 17 is incorporated and further claim 22 is a method claim corresponding to method claim 7 and is rejected for the reasons set forth in the rejection of claim 7.

As per claim 23 , the rejection of claim 17 is incorporated and further claim 23 is a method claim corresponding to method claim 8 and is rejected for the reasons set forth in the rejection of claim 8.

As per claim 29 , the rejection of claim 26 is incorporated and further claim 29 is a compiler claim corresponding to method claim 5 and is rejected for the reasons set forth in the rejection of claim 5.

As per claim 30 , the rejection of claim 26 is incorporated and further claim 30 is a compiler claim corresponding to method claim 6 and is rejected for the reasons set forth in the rejection of claim 6.

As per claim 31 , the rejection of claim 26 is incorporated and further claim 31 is a compiler claim corresponding to method claim 7 and is rejected for the reasons set forth in the rejection of claim 7.

As per claim 32 , the rejection of claim 26 is incorporated and further claim 32 is a compiler claim corresponding to method claim 8 and is rejected for the reasons set forth in the rejection of claim 8.

Conclusion

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andre R. Fowlkes whose telephone number is (703)305-8889. The examiner can normally be reached on Monday - Friday, 8:00am-4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703)305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

ARF

[illegible]

TUAN DAM
SUPERVISORY PATENT EXAMINER